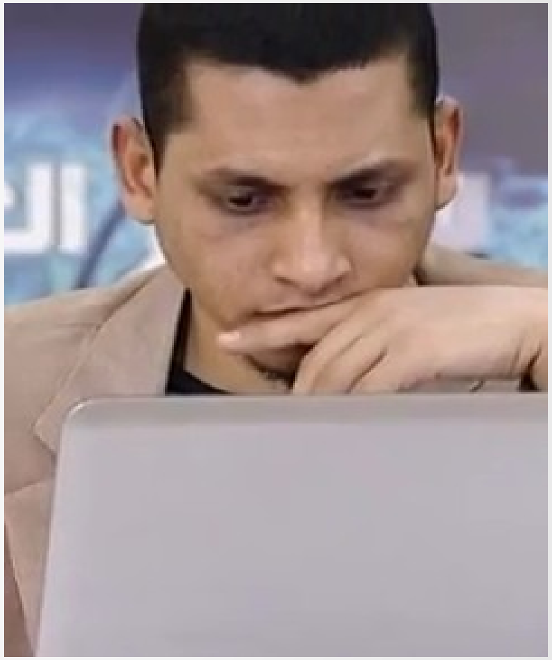


About ebda3 programming language

A little about me

- wael hasan mohammad.
- Graduated in 2010 from aswan faculty of engineering in egypt, electrical department, computer and systems branch.
- Lots of material about “control” systems (classical, modern, digital, plc … etc), and I hated them all except PLC because it have the programming things XD
- Almost nothing about software engineering !
- studied 4 different programming languages: VB, fortran 90, c++, java. But I used C# for both my personal projects and university projects (including graduation project which was developing a game using C# and XNA library).



- Worked for a while as an assistant researcher at C.A.I.R.O (Center for Artificial Intelligence and Robotics), at aswan faculty of sciences. Mainly in image processing stuff using c++ and OpenCV -_-
- loving programming + loving studying roots of any thing I love => language designing and implementing (interpreters + compilers) + operating systems design and implementation … etc (= a core programmer!).
- Big fan of linux (exactly: kubuntu, and I hate gnome and unity.... etc). Also big fan of linus torvalds.
- Using: “java + netbeans + shell scripting + batch files” in my projects.
- Loves cats, and hates dogs (so if you are an american you should LEAVE :P).
- Personal blog: afkar-abo-eyas.blogspot.com
- scientific blog: abo-eyas.blogspot.com

What is “ebda3” ?

the last “a” letter taking the place of the arabic letter ع which doesn't have an equivalent in English. And the name means “creativity”.

It is a professional arabic programming language, that aims to be the perfect high level general purpose programming language for **me** in different usage cases.

Reasons of designing and implementing ebda3:

- Personal usage; because I didn't find the programming language that have all the characteristics that I like (really tough ones),
- Public usage; because I think that my rules of designing ebda3 made it great for other people too (that can write arabic XD).

Was influenced by

(Influence may be in a positive way, or it can be in a negative way, or both).

C#

Java

Matlab

C++

Visual basic

F#

ج

Fortran

Ada

Pascal

python

Eiffel

C

And other interesting languages :)

simplified rules of ebda3 design

"Component" means "any rule, mark, key word … etc".

- If you need a "component": you "must" add it.
- If you don't need a "component": you "must" remove it.
- If you can combine two or more components in one powerful component: you "must" combine them.
- If you can simplify a component: you must simplify it.

Ebda3 characteristics

simple:

number of its rules is as minimum and as easy as it can be, so that a medium size book can cover "all" its specification. While we see that other "powerful" and "professional" programming languages have at least twice the size of ebda3 !

This is not minimalist as in gnome, it is a minimalist of having the most powerful components by adding the least number of rules in the language. So you still have the needed power to work efficiently.

there are programming languages that have at least 3 times size as ebda3 like c++, object-pascal, and visual basic.NET.

Ebda3 characteristics

simple:

number of its rules is as minimum and as easy as it can be, so that a medium size book can cover "all" its specification. While we see that other "powerful" and "professional" programming languages have at least twice the size of ebda3 !

This is not minimalist as in gnome, it is a minimalist of having the most powerful components by adding the least number of rules in the language. So you still have the needed power to work efficiently.

there are programming languages that have at least 3 times size as ebda3 like c++, object-pascal, and visual basic.NET.

easy and educational:

it can be used as an educational programming language for children easily; because that was a "design criteria" when the language was designed.

In the same time most of the famous languages can't be used in education without a great pain; because educational purposes weren't a real design criteria for the designers of those languages.

Easiness in ebda3 design can be seen (just examples) in:

- type system:

  which is very simple; ebda3 has a hybrid type system but without the complexity of mexing sizes with types (for example: having a lot of types describing NUMBERS but with different sizes, like int and float... etc).

  So it has the three basic types (رقم) to define numbers that can be a floating point or integers, and of any size. And there is "نص" type that handles strings regardless of number of characters inside them. And it has the type "منطق" that is equivalent to boolean.

```
نص أ = " مرحباً "
رقم س= 20 ص
منطق ع = صحيح  ل = خطأ ك
حر ج = س+2
ج = " هذا نص ! "
```

- **there is no components that looks alike each other with small differences (like functions and procedures), and every component can do more than one job very easily and without confusion wether to use that component or the other.**

Hybrid (i.e: multi-paradigm):

it has characteristics that belongs to both: procedural languages, and object oriented languages. And tries to get the best of both of them.

And this means that you can use the paradigm that fits with the application you are building, but take care that you can only use procedural paradigm inside the main script, and not in the package files (looks so much like ".java" files).

Powerful:

because that was also a "design criteria", so you can see that alot of components of the language (or "rules") can replace more than one rule in other languages.

You can see that conditional statemen "لو" which looks like "if" statement can be used in defferent ways so that it is enough and we don't need other shapes of conditional statements.

```
لو س = ص :
    أكتب.سطر("س يساوي ص")
أما لو س في {من 1 إلي 12}:
    أكتب.سطر("س في المجموعة من 1 إلي 12")
أما لو س> 12 و س<20:
    أكتب.سطر("س بين الـ 12 و الـ 20")
غيره :
    أكتب.سطر("س خارج الاحتمالات الممنوحة")
```

- Also there is the loop expression "بينما" which replaces the "for" and "do … while" and "foreach" loops.

```
بينما م في {من البداية إلي النهاية بقيمة 5}:
    أكتب.سطر("م = "+ م)
```

```
بينما م في {1 2 5 7 10}:
    أكتب.سطر("م = "+م)
```

```
بينما صحيح:
    \ نفذ أمرأ ما /
    لو شرط :
        أخرج()
```

- Of course that helped a lot in making the size of the language smaller; because instead of having more than one rule we saw that one is really enough, so we managed to make size smaller without Sacrificing power.
- examples for that:
  - structs + classes + interfaces = صنف
  - functions + procedures + delegates + properties = إجراء
  - arrays + hashtables = الجداول

secure and clean:

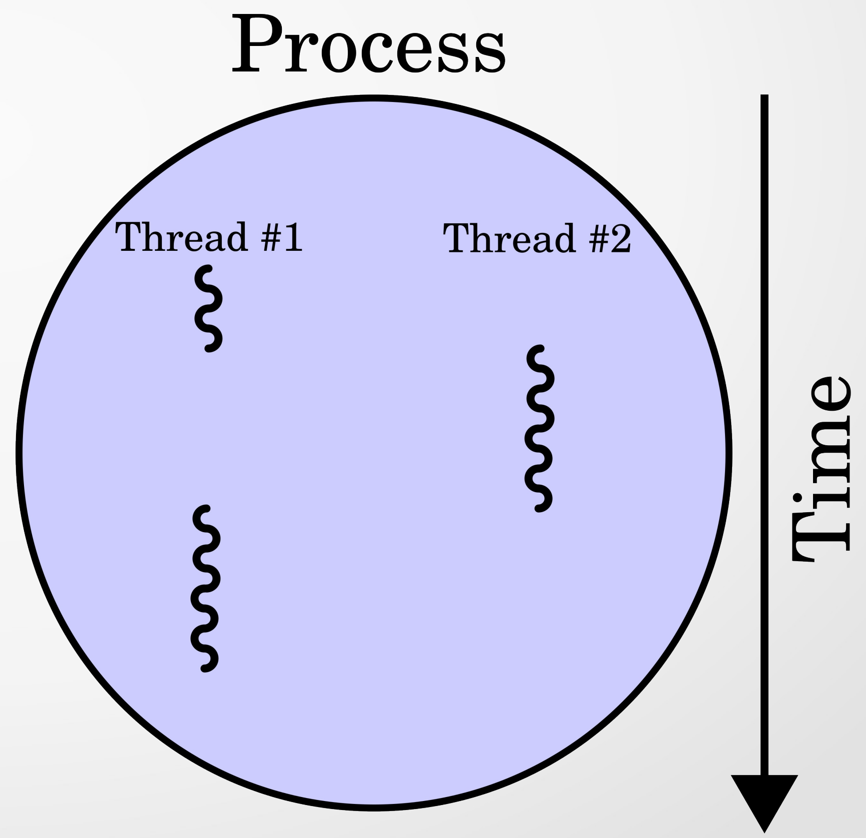
it choses the best rules for the programmer to work with, and it tries (as it can) to use "safe" components to handle tasks, so you are not going to see "pointers" in ebda3, but you can do the same jobs in a different (yet safe) way.

```
void CvtImg2Dta(IplImage *img, double *slice)
{
    for( int y=0; y< img->height; y++ )// taking care of rows indexing
    {
        for( int x=0; x< img->width; x++ )// taking care of columns indexing
        {
            slice[img->nChannels*x+y*img->width*img->nChannels] =
                    (double)img->imageData[img->nChannels*x+y*img->widthStep]/255.0;
            if(img->nChannels==3)
            {
                slice[(3*x)+(3*y*img->width) +1] =
                    (double)img->imageData[(3*x)+(y*img->widthStep) +1]/255.0;
                slice[(3*x)+(3*y*img->width) +2] =
                    (double)img->imageData[(3*x)+(y*img->widthStep) +2]/255.0;
            }
        }
    }
}
```

parallel executing:

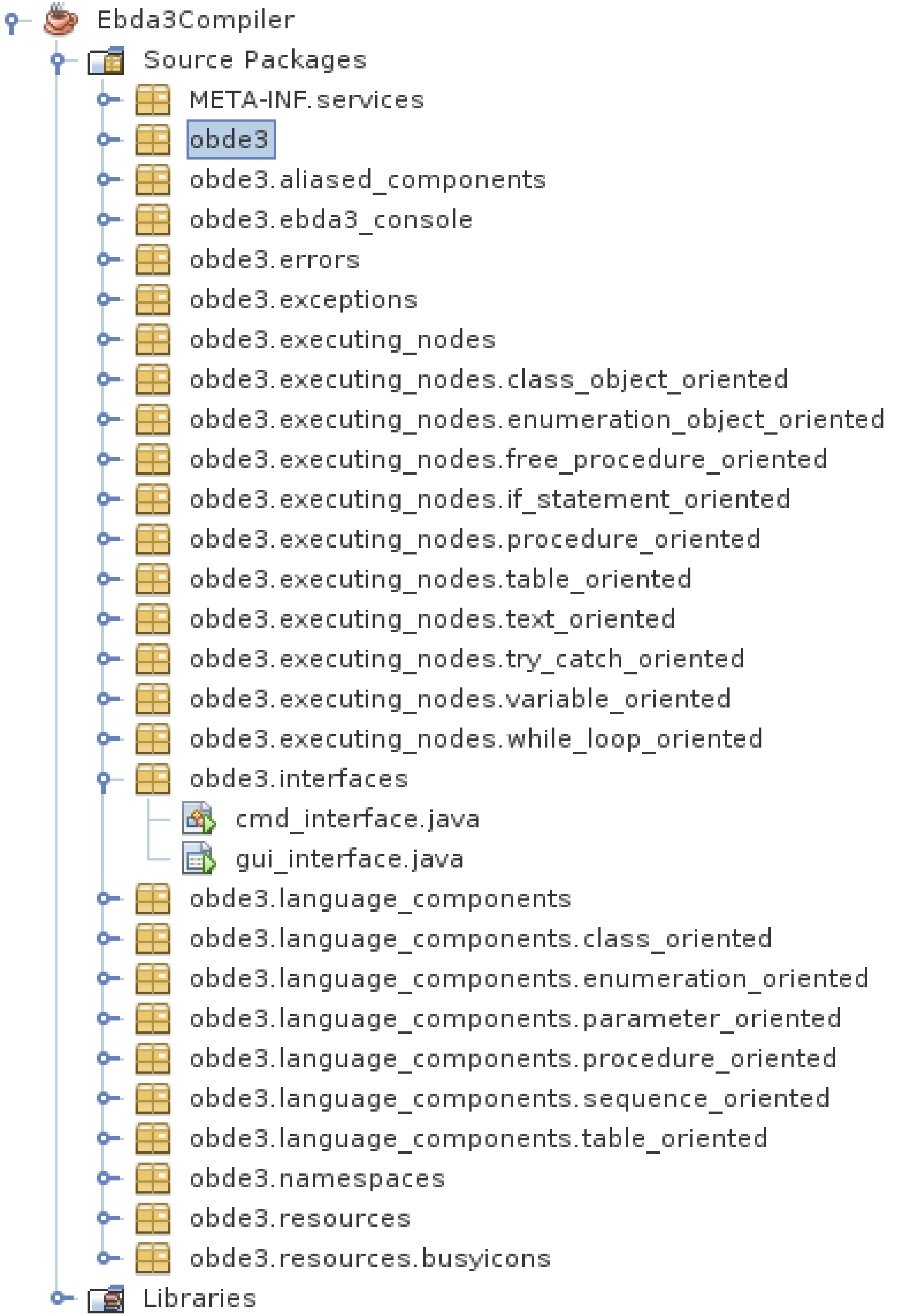
small and easy syntax for working in multi-threading cases. So you can concentrate in your algorithms, and implement them easily.

but this isn't implemented until now in the standard interpreter of the language (which is called "obde3").



interpreted and compiled:

so it can generate managed code or native code with the same source code and same libraries, (this isn't implemented until now in obde3).



free (as in "freedom", and as in "free lunch"):

the standard interpreter of the language and its standard library, and all other tools (standard IDE, standard debugger, .. etc) are (or will soon be) open source, and they are all free and you don't need to buy them.

Stable:

there is no intention to modify its specifications widely, and new changes will "always" be small even if they have great influences on easiness, also they won't make your old working code go "broken" in the new versions.

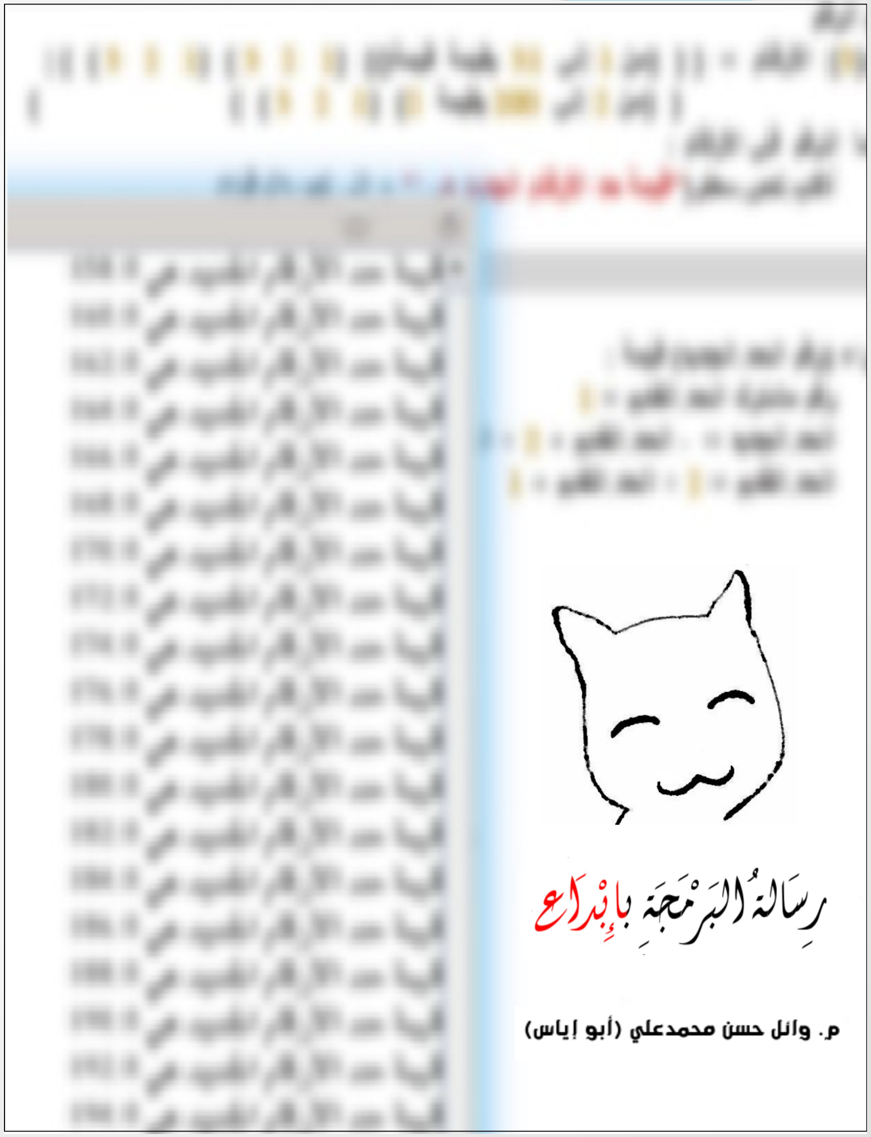


Is it just ebda3 ?

- No; ebda3 is just the “perfect tool” for me to use in my projects, which are:
  - powerful IDE and programming tools.
  - powerful operating system, and powerful desktop environment that concentrates on “contents” instead of “tools”.
  - Powerful file system that can moderate data more powerfully.
  - Writing books about software that uses ebda3 in teaching.

message of creatively programming

official book of the project, which talks about every things belongs to it (an arabic book called "message of creatively programming رسـالة البرمجة بإبداع"):



Links

Official blog of the project:

http://ebda3lang.blogspot.com/

links for downloading "message of creatively programming رسالة البرمجة بإبداع":

http://ebda3lang.blogspot.com/p/blog-page.html

links for downloading the standard interpreter "obde3" can be found in:

http://ebda3lang.blogspot.com/p/blog-page_5.html

links for downloading the lightweight standard IDE "andalos" can be found in:

http://ebda3lang.blogspot.com/p/blog-page_17.html

Thank you for listening XD

